```
import pandas as pd

# Load the data

df = pd.read_csv(r"C:\Users\nikhi\OneDrive\Desktop\online_advertising_performance_data.csv")

df.dropna(axis=1, how='all', inplace=True)


df['month'] = df['month'].astype('category')

df['campaign_number'] = df['campaign_number'].astype('category')

df['user_engagement'] = df['user_engagement'].astype('category')

df['banner'] = df['banner'].astype('category')

df['placement'] = df['placement'].astype('category')

df['day'] = df['day'].astype('int')


# 1. Overall trend in user engagement throughout the campaign period

plt.figure(figsize=(12, 6))

sns.countplot(x='month', hue='user_engagement', data=df,

            palette='viridis')

plt.title('User Engagement Trend by Month')

plt.xlabel('Month')

plt.ylabel('Count')

plt.show()


# 2. Impact of ad size (banner) on the number of clicks generated

plt.figure(figsize=(14, 7))

sns.barplot(x='banner', y='clicks', data=df, palette='muted')

plt.xticks(rotation=45, ha='right')

plt.title('Impact of Banner Size on Clicks')

plt.xlabel('Banner Size')

plt.ylabel('Clicks')

plt.tight_layout()  # Adjust layout

plt.show()
```

```python
# 3. Publisher spaces (placements) yielding the highest number of displays and clicks
plt.figure(figsize=(14, 7))
sns.barplot(x='placement', y='displays', data=df,
            palette='plasma')
plt.xticks(rotation=45, ha='right')
plt.title('Displays by Placement')
plt.xlabel('Placement')
plt.ylabel('Displays')
plt.tight_layout()
plt.show()

plt.figure(figsize=(14, 7))
sns.barplot(x='placement', y='clicks', data=df,
            palette='magma')
plt.xticks(rotation=45, ha='right')
plt.title('Clicks by Placement')
plt.xlabel('Placement')
plt.ylabel('Clicks')
plt.tight_layout()
plt.show()

# 4. Correlation between cost of serving ads and revenue generated from clicks
plt.figure(figsize=(8, 6))
sns.scatterplot(x='cost', y='revenue', data=df, alpha=0.5)
plt.title('Correlation between Cost and Revenue')
plt.xlabel('Cost')
plt.ylabel('Revenue')
plt.show()

# 5. Average revenue generated per click
df['revenue_per_click'] = df['revenue'] / df['clicks']
```

```
df['revenue_per_click'].fillna(0, inplace=True)

average_revenue_per_click = df['revenue_per_click'].mean()
print(f"Average Revenue Per Click: {average_revenue_per_click:.2f}")

# 6. Campaigns with the highest post-click conversion rates
df['post_click_conversion_rate'] = df['post_click_conversions'] / df['clicks']
df['post_click_conversion_rate'].fillna(0, inplace=True)

campaign_conversion_rates = df.groupby('campaign_number')[
    'post_click_conversion_rate'].mean().sort_values(ascending=False)
print("Campaign Post-Click Conversion Rates:\n", campaign_conversion_rates)

# 7. Trends or patterns in post-click sales amounts over time
plt.figure(figsize=(12, 6))
sns.lineplot(x='day', y='post_click_sales_amount', data=df)
plt.title('Post-Click Sales Amount Over Time')
plt.xlabel('Day')
plt.ylabel('Post-Click Sales Amount')
plt.show()

# 8. Level of user engagement across different banner sizes
plt.figure(figsize=(12, 6))
sns.countplot(x='banner', hue='user_engagement', data=df)
plt.xticks(rotation=45, ha='right')
plt.title('User Engagement by Banner Size')
plt.xlabel('Banner Size')
plt.ylabel('Count')
plt.tight_layout()
plt.show()
```

```
# 9. Placement types resulting in the highest post-click conversion rates
placement_conversion_rates = df.groupby('placement')[
    'post_click_conversion_rate'].mean().sort_values(ascending=False)
print("Placement Post-Click Conversion Rates:\n", placement_conversion_rates)

# 10. Seasonal patterns or fluctuations in displays and clicks throughout the campaign period
plt.figure(figsize=(12, 6))
sns.lineplot(x='month', y='displays', data=df,
             label='Displays', marker='o')  # added marker
sns.lineplot(x='month', y='clicks', data=df, label='Clicks', marker='o')
plt.title('Displays and Clicks Throughout Campaign Period')
plt.xlabel('Month')
plt.ylabel('Count')
plt.legend()
plt.show()

# 11. Correlation between user engagement levels and the revenue generated
engagement_revenue = df.groupby('user_engagement')['revenue'].mean()
print("Average Revenue by User Engagement:\n", engagement_revenue)

# 12. Outliers in terms of cost, clicks, or revenue
plt.figure(figsize=(15, 5))

plt.subplot(1, 3, 1)
sns.boxplot(y=df['cost'])
plt.title('Cost Distribution')

plt.subplot(1, 3, 2)
sns.boxplot(y=df['clicks'])
plt.title('Clicks Distribution')
```

```
plt.subplot(1, 3, 3)
sns.boxplot(y=df['revenue'])
plt.title('Revenue Distribution')

plt.tight_layout()
plt.show()

# 13. Effectiveness of campaigns based on ad size and placement type
plt.figure(figsize=(14, 7))
sns.barplot(x='banner', y='post_click_conversions', hue='placement', data=df)
plt.xticks(rotation=45, ha='right')
plt.title('Post-Click Conversions by Banner and Placement')
plt.xlabel('Banner Size')
plt.ylabel('Post-Click Conversions')
plt.tight_layout()
plt.show()

# 14. Specific campaigns or banner sizes that consistently outperform others in terms of ROI

df['ROI'] = (df['revenue'] - df['cost']) / df['cost']
df['ROI'].replace([float('inf'), float('-inf')], 0,
                  inplace=True)
df['ROI'].fillna(0, inplace=True)  #

campaign_roi = df.groupby('campaign_number')['ROI'].mean().sort_values(
    ascending=False)
print("Campaign ROI:\n", campaign_roi)

banner_roi = df.groupby('banner')['ROI'].mean().sort_values(ascending=False)
print("Banner ROI:\n", banner_roi)
```

```
# 15. Distribution of post-click conversions across different placement types
plt.figure(figsize=(12, 6))
sns.boxplot(x='placement', y='post_click_conversions', data=df)
plt.xticks(rotation=45, ha='right')
plt.title('Distribution of Post-Click Conversions Across Placements')
plt.xlabel('Placement')
plt.ylabel('Post-Click Conversions')
plt.tight_layout()
plt.show()

# 16. Differences in user engagement levels between weekdays and weekends

# Extract weekday (Monday=0, Sunday=6)
df['weekday'] = pd.to_datetime(df['day'], unit='D', origin=pd.Timestamp(
    '2023-03-31'))
df['weekday'] = df['weekday'].dt.weekday

# Map to 'Weekend' and 'Weekday'
df['day_type'] = df['weekday'].apply(lambda x: 'Weekend' if x >= 5 else 'Weekday')

plt.figure(figsize=(8, 6))
sns.countplot(x='day_type', hue='user_engagement', data=df)
plt.title('User Engagement Levels - Weekdays vs. Weekends')
plt.xlabel('Day Type')
plt.ylabel('Count')
plt.show()

# 17. Cost per click (CPC) across different campaigns and banner sizes
df['CPC'] = df['cost'] / df['clicks']
df['CPC'].replace([float('inf'), float('-inf')], 0,
                  inplace=True)
```

```
df['CPC'].fillna(0, inplace=True)


campaign_cpc = df.groupby('campaign_number')['CPC'].mean()
print("Average CPC by Campaign:\n", campaign_cpc)


banner_cpc = df.groupby('banner')['CPC'].mean()
print("Average CPC by Banner:\n", banner_cpc)


# 18. Campaigns or placements that are particularly cost-effective in terms of generating post-click conversions
df['cost_per_conversion'] = df['cost'] / df['post_click_conversions']
df['cost_per_conversion'].replace([float('inf'), float('-inf')], 0,
                                  inplace=True)
df['cost_per_conversion'].fillna(0, inplace=True)


campaign_cost_per_conversion = df.groupby('campaign_number')[
    'cost_per_conversion'].mean().sort_values()
print("Campaign Cost Per Conversion:\n", campaign_cost_per_conversion)


placement_cost_per_conversion = df.groupby('placement')[
    'cost_per_conversion'].mean().sort_values()
print("Placement Cost Per Conversion:\n", placement_cost_per_conversion)


# 19. Trends or patterns in post-click conversion rates based on the day of the week
plt.figure(figsize=(12, 6))
sns.lineplot(x='weekday', y='post_click_conversion_rate', data=df)
plt.xticks(range(7),
           ['Mon', 'Tue', 'Wed', 'Thu', 'Fri', 'Sat', 'Sun'])
plt.title('Post-Click Conversion Rates by Day of the Week')
plt.xlabel('Day of the Week')
plt.ylabel('Post-Click Conversion Rate')
```

```
plt.show()
```

```
# 20. Effectiveness of campaigns throughout different user engagement types in terms of post-click conversions

plt.figure(figsize=(10, 6))

sns.barplot(x='user_engagement', y='post_click_conversions', data=df)

plt.title('Post-Click Conversions by User Engagement')

plt.xlabel('User Engagement')

plt.ylabel('Post-Click Conversions')

plt.show()
```